User Manual

ASCII

Handshake Mechanism

Hilscher Gesellschaft für Systemautomation mbH

www.hilscher.com

DOC100213UM03EN | Revision 3 | English | 2011-05 | Released | Public

Table of Contents

1 INTRODUCTION 3
1.1 Introduction to this Manual 3
1.1.1 List of Revisions 3
1.1.2 Reference to Firmware and Software 4
1.1.3 Conventions in this Manual 5
1.2 Legal Notes 6
1.2.1 Copyright 6
1.2.2 Important Notes 6
1.2.3 Exclusion of Liability 7
1.2.4 Warranty 7
1.2.5 Export Regulations 8
1.2.6 Registered Trademarks 8
2 ASCII 9
2.1 Introduction 9
2.2 I/O Data Structure for the Transfer to and from the Control Unit 10
2.2.1 Structure for Output - Data from the Control Unit 10
2.2.2 Structure for Input - Data to the Control Unit 10
2.3 Handshake and Initialization of the I/O Communication 11
2.3.1 Structure of the Synchronization Register in the I/O Data 13
2.3.2 Initializing of the Communication 15
2.3.3 Acknowledgment of the Processing between the Superordinated Control and ASCII 16
3 ERROR CODES 19
4 APPENDIX 20
4.1 List of Figures 20
4.2 List of Tables 20
4.3 Contacts 21

© Hilscher, 2010-2011

1 Introduction

1.1 Introduction to this Manual

This manual provides information about the handshake mechanism for serial ASCII communication.

The ASCII protocol for serial data transfer is used on netTAP NT 50, netTAP NT 100 and netBRICK NB 100 gateway devices.

1.1.1 List of Revisions

Index	Date	Chapter	Revision
1	2010-02-19	All	Created
2	2010-03-18	2.3	Description expanded in section *Handshake and Initialization of the I/O Communication*: Pair of bits for send respectively receive an ASCII telegram and *Relationship between send/receive ASCII Telegram and ASCII Operating Mode*
3	2011-05-30	1.1.2	Section *Reference to Firmware and Software*: netTAP NT 50 and netBRICK NB 100 added

Table 1: List of Revisions

© Hilscher, 2010-2011

1.1.2 Reference to Firmware and Software

Firmware

Firmware File	Firmware Version
N5ASCxxx.NXF repectively N5xxxASC.NXF netTAP NT 50 firmware with ASCII protocol	from 1.0.x.x
NTxxxASC.NXF netTAP NT 100 firmware with ASCII protocol	from 1.3.x.x
NBASCxxx.NXF netBRICK NB 100 Firmware with ASCII protocol	from 1.4.x.x

Table 2: Reference to Firmware

Software

Software	Software Version
SYCONnet netX setup.exe	from 1.310.x.x

Table 3: Reference to Software

© Hilscher, 2010-2011

1.1.3 Conventions in this Manual

Operation instructions, a result of an operation step or notes are marked as follows:

Operation Instructions:

➢ <instruction>

Or

1. <instruction>
2. <instruction>

Results:

➮ <result>

Notes:

Important: <important note>

Note: <note>

<note, were to find further information>

© Hilscher, 2010-2011

1.2 Legal Notes

1.2.1 Copyright

© Hilscher, 2010-2011, Hilscher Gesellschaft für Systemautomation mbH

All rights reserved.

The images, photographs and texts in the accompanying material (user manual, accompanying texts, documentation, etc.) are protected by German and international copyright law as well as international trade and protection provisions. You are not authorized to duplicate these in whole or in part using technical or mechanical methods (printing, photocopying or other methods), to manipulate or transfer using electronic systems without prior written consent. You are not permitted to make changes to copyright notices, markings, trademarks or ownership declarations. The included diagrams do not take the patent situation into account. The company names and product descriptions included in this document may be trademarks or brands of the respective owners and may be trademarked or patented. Any form of further use requires the explicit consent of the respective rights owner.

1.2.2 Important Notes

The user manual, accompanying texts and the documentation were created for the use of the products by qualified experts, however, errors cannot be ruled out. For this reason, no guarantee can be made and neither juristic responsibility for erroneous information nor any liability can be assumed. Descriptions, accompanying texts and documentation included in the user manual do not present a guarantee nor any information about proper use as stipulated in the contract or a warranted feature. It cannot be ruled out that the user manual, the accompanying texts and the documentation do not correspond exactly to the described features, standards or other data of the delivered product. No warranty or guarantee regarding the correctness or accuracy of the information is assumed.

We reserve the right to change our products and their specification as well as related user manuals, accompanying texts and documentation at all times and without advance notice, without obligation to report the change. Changes will be included in future manuals and do not constitute any obligations. There is no entitlement to revisions of delivered documents. The manual delivered with the product applies.

Hilscher Gesellschaft für Systemautomation mbH is not liable under any circumstances for direct, indirect, incidental or follow-on damage or loss of earnings resulting from the use of the information contained in this publication.

1.2.3 Exclusion of Liability

The software was produced and tested with utmost care by Hilscher Gesellschaft für Systemautomation mbH and is made available as is. No warranty can be assumed for the performance and flawlessness of the software for all usage conditions and cases and for the results produced when utilized by the user. Liability for any damages that may result from the use of the hardware or software or related documents, is limited to cases of intent or grossly negligent violation of significant contractual obligations. Indemnity claims for the violation of significant contractual obligations are limited to damages that are foreseeable and typical for this type of contract.

It is strictly prohibited to use the software in the following areas:

- for military purposes or in weapon systems;
- for the design, construction, maintenance or operation of nuclear facilities;
- in air traffic control systems, air traffic or air traffic communication systems;
- in life support systems;
- in systems in which failures in the software could lead to personal injury or injuries leading to death.

We inform you that the software was not developed for use in dangerous environments requiring fail-proof control mechanisms. Use of the software in such an environment occurs at your own risk. No liability is assumed for damages or losses due to unauthorized use.

1.2.4 Warranty

Although the hardware and software was developed with utmost care and tested intensively, Hilscher Gesellschaft für Systemautomation mbH does not guarantee its suitability for any purpose not confirmed in writing. It cannot be guaranteed that the hardware and software will meet your requirements, that the use of the software operates without interruption and that the software is free of errors. No guarantee is made regarding infringements, violations of patents, rights of ownership or the freedom from interference by third parties. No additional guarantees or assurances are made regarding marketability, freedom of defect of title, integration or usability for certain purposes unless they are required in accordance with the law and cannot be limited. Warranty claims are limited to the right to claim rectification.

© Hilscher, 2010-2011

1.2.5 Export Regulations

The delivered product (including the technical data) is subject to export or import laws as well as the associated regulations of different counters, in particular those of Germany and the USA. The software may not be exported to countries where this is prohibited by the United States Export Administration Act and its additional provisions. You are obligated to comply with the regulations at your personal responsibility. We wish to inform you that you may require permission from state authorities to export, re-export or import the product.

1.2.6 Registered Trademarks

Windows® 2000, Windows® XP, Windows® Vista and Windows® 7 are registered trademarks of Microsoft Corporation.

All other mentioned trademarks are property of their respective legal owners.

© Hilscher, 2010-2011

2 ASCII

2.1 Introduction

The superordinated control unit communicates with a cyclically running protocol with the netTAP respectively with the netBRICK device. However, ASCII works acyclically (job oriented). A handshake mechanism is necessary to connect both communication types. Therefore synchronization registers are included in the transfer structure to the ASCII protocol

The following figure shows that data from the ASCII protocol can be mapped to the protocol on port X2 and data from port X2 can be mapped to the ASCII protocol. In this mapping, the synchronization registers are included as well as the user data.

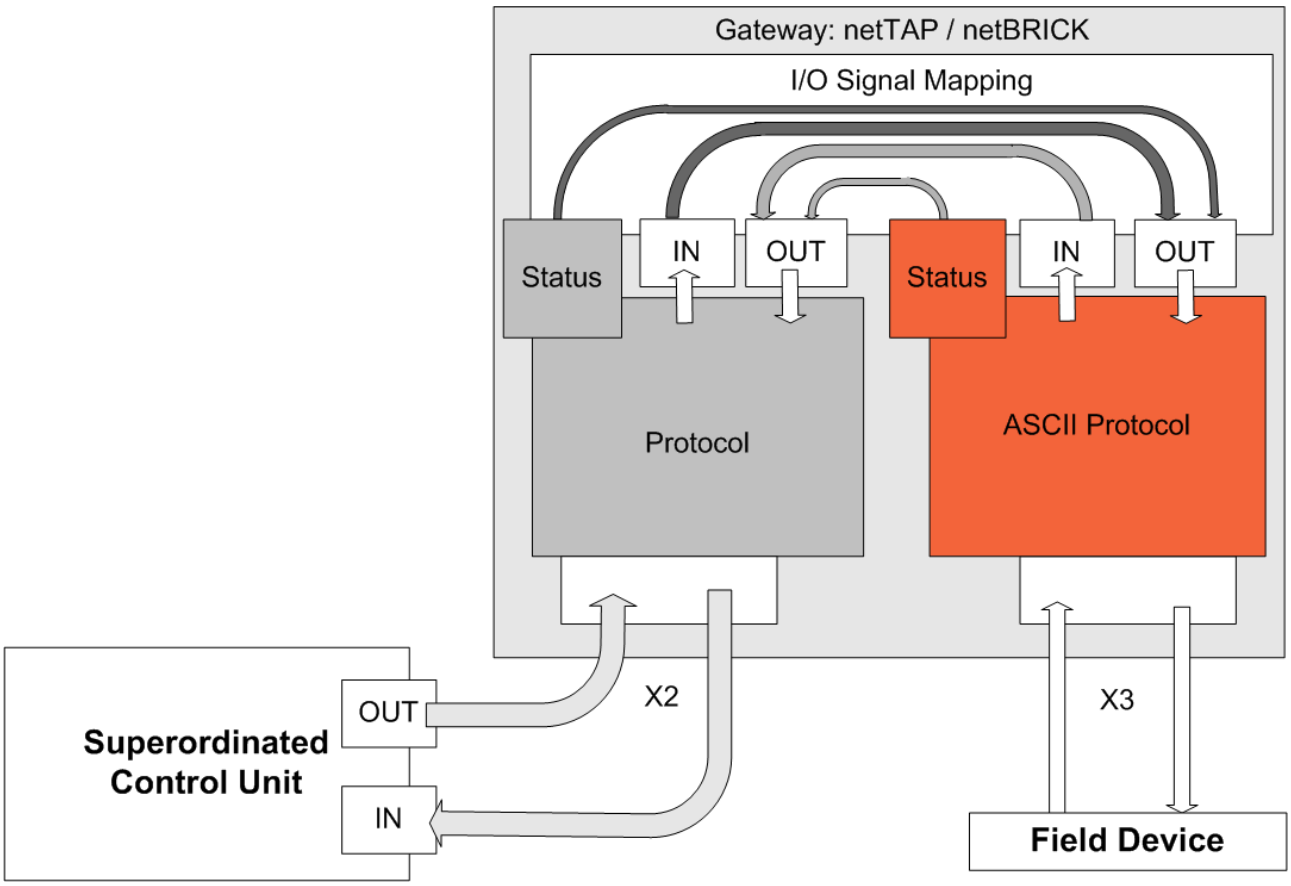

Figure 1: Principal Mapping of the Data between the Protocols

The signal mapping of the data is configured with the configuration software SYCON.net.

© Hilscher, 2010-2011

2.2 I/O Data Structure for the Transfer to and from the Control Unit

2.2.1 Structure for Output - Data from the Control Unit

Data Type	Signal	Signal for mapping in SYCON.net
UINT32	Output synchronization register	Application handshake flags
UINT32	Amount of user data (counted in bytes) to be transferred	Byte count of OutData
Array of Bytes[0 … 1023]	Output User data for the superordinated control unit and send user data for ASCII	OutData.UNSIGNED8_0000 OutData.UNSIGNED8_0001 … OutData.UNSIGNED8_1023

Table 4: ASCII – Structure for Output – Data from the Control Unit

2.2.2 Structure for Input - Data to the Control Unit

Data Type	Signal	Signal for mapping in SYCON.net
UINT32	Input synchronization register	Protocol handshake flags
UINT32	Amount of user data (counted in bytes) o be transferred	Byte count of InData
UINT32	Error register to transfer error information about receive errors	Error code in case of receive error
UINT32	Error register to transfer error information about transmit errors	Error code in case of transmit error
Array of Bytes[0 … 1023]	Receive user data for ASCII and input user data for the superordinated control unit	InData.UNSIGNED8_0000 InData.UNSIGNED8_0001 … InData.UNSIGNED8_1023

Table 5: ASCII – Structure for Input – Data to the Control Unit

© Hilscher, 2010-2011

2.3 Handshake and Initialization of the I/O Communication

The data transfer is between the control unit and the ASCII protocol is organized by a transfer method within the I/O data transfer memory.

The basic idea of this method is: for each action, a pair of bits is used in both synchronization registers. One bit is used to request an action, and the other is used to acknowledge the action. One is located in the input synchronization register, and the other in the output synchronization register.

One action is requested by setting the command bit unequal to the acknowledge bit. The other side acknowledges this request by setting the acknowledge bit equal to the command bit.

Send ASCII Telegram

One pair of bits is used to send an ASCII telegram. The superordinated control unit controls that an ASCII telegram has to be send. The bits of this pair of bits have the name APP_HS_TX_CMD and PROT_HS_TX_ACK and are described in the following sections:

- APP_HS_TX_CMD is described in Table 7: ASCII – Synchronization Register to ASCII on page 13
- PROT_HS_TX_ACK is described in Table 8: ASCII – Synchronization Register to the Superordinated Control Unit on page 14
- The usage is described in section Superordinated Control Unit to ASCII on page 16

Receive ASCII Telegram

An other pair of bits is used to receive an ASCII telegram. The superordinated control unit recognizes that an ASCII telegram was received. The bits of this pair of bits have the name APP_HS_RX_ACK and PROT_HS_RX_CMD and are described in the following sections:

- APP_HS_RX_ACK is described in Table 7: ASCII – Synchronization Register to ASCII on page 13
- PROT_HS_RX_CMD is described in Table 8: ASCII – Synchronization Register to the Superordinated Control Unit on page 14
- The usage is described in section ASCII to the superordinated Control Unit on page 17

© Hilscher, 2010-2011

Relationship between send/receive ASCII Telegram and ASCII Operating Mode

ASCII offers four operating modes. The following table shows the relationship between send/receive an ASCII telegram and the four operating modes from the viewpoint of the superordinated control unit:

Operating Mode	Pair of bits	Description
'Receive Only' Mode	APP_HS_TX_CMD and PROT_HS_TX_ACK	Superordinated Control Unit to ASCII on page 16
'Send Only' Mode	APP_HS_RX_ACK and PROT_HS_RX_CMD	ASCII to the superordinated Control Unit on page 17
Client Mode (first send, then receive)	APP_HS_TX_CMD and PROT_HS_TX_ACK	Superordinated Control Unit to ASCII on page 16
	APP_HS_RX_ACK and PROT_HS_RX_CMD	ASCII to the superordinated Control Unit on page 17
Server Mode (first receive, then send)	APP_HS_RX_ACK and PROT_HS_RX_CMD	ASCII to the superordinated Control Unit on page 17
	APP_HS_TX_CMD and PROT_HS_TX_ACK	Superordinated Control Unit to ASCII on page 16

Table 6: Relationship between send/receive ASCII Telegram and ASCII Operating Mode

© Hilscher, 2010-2011

2.3.1 Structure of the Synchronization Register in the I/O Data

2.3.1.1 Superordinated Control Unit to ASCII

Structure of the synchronization register of the control unit to ASCII:

Bit No.	Name and Description
0	**APP_HS_TX_CMD** Command from the control unit to send output data to ASCII. Is checked by ASCII automatically.
1	**APP_HS_RX_ACK** Acknowledge received input data in the control unit. The bit is checked automatically within ASCII.
2 … 5	Not used and reserved
6	**APP_HS_TX_ENABLE_CMD** Enable the output user data transfer from the control unit to ASCII. If this bit is not set, then ASCII will not evaluate requested commands via the bit **APP_HS_TX_CMD**.
7	**APP_HS_RX_ENABLE_CMD** Enable the input user data transfer from ASCII to the control unit. If this bit is not set, then ASCII can not requested commands via the bit **APP_HS_RX_CMD**.
8 … 31	Not used and reserved

Table 7: ASCII – Synchronization Register to ASCII

© Hilscher, 2010-2011

2.3.1.2 ASCII to Superordinated Control Unit

Structure of the synchronization register from ASCII to the control unit:

Bit No.	Name and Description
0	**PROT_HS_TX_ACK** Acknowledge bit from ASCII to the superordinated control unit for sent output data. Is used by ASCII automatically.
1	**PROT_HS_RX_CMD** Command for input user data from ASCII to the superordinated control unit. The bit is checked automatically within ASCII.
2	Not used and reserved
3	**PROT_HS_RUN_IND** The ASCII protocol signals 'Ready' and the end of initialization. This bit is set independently from the command bit **APP_HS_RX_ENABLE_CMD**.
4	**PROT_HS_TX_ERROR_IND** To indicate, if a send error has occurred. Is set by the ASCII protocol. This bit is set back to zero (0) by the ASCII protocol with the next error free send telegram. 0: No send error 1: Send error occurred. The error code is written by the ASCII protocol into the “Error register to transfer error information about receive errors“. The error codes are described in section *Error Codes* on page 19.
5	**PROT_HS_RX_ERROR_IND** To indicate, if a receive error has occurred. Is set by the ASCII protocol. This bit is set back to zero (0) with the next error free receive telegram. 0: No receive error 1: Receive error. The error code is written by the ASCII protocol into the “Error register to transfer error information about receive errors“. The error codes are described in section *Error Codes* on page 19.
6	**PROT_HS_TX_ENABLE_ACK** Acknowledges the enabling of output data transfer from the control unit to ASCII. This bit is controlled from ASCII automatically.
7	**PROT_HS_RX_ENABLE_ACK** Acknowledges the enabling of input data transfer from ASCII to the control unit. This bit is controlled from ASCII automatically.
8 … 31	Not used and reserved

Table 8: ASCII – Synchronization Register to the Superordinated Control Unit

© Hilscher, 2010-2011

2.3.2 Initializing of the Communication

Start of the communication

Step	Action: **Start of the communication, initialization is done by the superordinated control unit.**	**Handshake-send byte of the superordinated control unit** 7 6 5 4 3 2 1 0	**Handshake-receive byte of the superordinated control unit** 7 6 5 4 3 2 1 0
0	Memory after device reset. ASCII signals 'not ready'.	0 0 x x x x 0 0	0 0 0 0 0 0 0 0
1	ASCII signals 'ready'.	0 0 x x x x 0 0	0 0 0 0 1 0 0 0
2	The superordinated control unit starts the communication with ASCII. Setting bit 6 and 7 allows ASCII to communicate with the superordinated control unit.	1 1 x x x x 0 0	0 0 0 0 1 0 0 0
3	ASCII receives the handshake flags from the superordinated control unit. These release the following actions: The send direction to the superordinated control unit is enabled based on bit 6. The receive direction for ASCII is enabled based on bit 7.	1 1 x x x x 0 0	
4	The receipt is acknowledges by ASCII. The data transfer to the superordinated control unit can start.		1 1 0 0 1 0 0 0
5	After the superordinated control unit has received the acknowledgment from ASCII for 'send and receive ready' it can send data to ASCII.		1 1 0 0 1 0 0 0

Table 9: ASCII – Initializing of the Communication

Note: After enabling the communication the communication can start any time.

© Hilscher, 2010-2011

2.3.3 Acknowledgment of the Processing between the Superordinated Control and ASCII

An acknowledgment of the receipt is expected (in the corresponding synchronization register) from the receiver for each data transfer from ASCII to the superordinated control unit and visa versa. As long as this receive acknowledgment is not available, no further data can be send to the receiver.

This handshake procedure is described in the following section for both directions.

Note: In the following tables an "x" marks an undefined bit position and an "X" marks a defined but not relevant bit position.

2.3.3.1 Superordinated Control Unit to ASCII

The value of a bit marked with "x" don't cares.

Step	Action: The superordinated control unit sends data to ASCII	Handshake-send byte of the superordinated control unit / receive byte of the gateway 7 6 5 4 3 2 1 0	Handshake-receive byte of the superordinated control unit / send byte of the gateway 7 6 5 4 3 2 1 0
0	State before beginning to send (and after initialization). Handshake bit 0 in the receive and send buffer has the same value. → It is possible to send data to ASCII.	1 1 x x x x X 0	1 1 x x 1 x X 0
1	The superordinated control unit has provided user data in its send buffer to be send and sets handshake bit 0.	1 1 x x x x X 1	
2	The data is transferred to ASCII	1 1 x x x x X 1	
3	As long as bit 0 of the handshake in the send and receive buffer is unequal, it is not allowed for the superordinated control unit to send new user data to the gateway.	1 1 x x x x X 1	1 1 x x 1 x X 0
4	ASCII recognizes that bit 0 of the handshake in the send and receive buffer is unequal, that new user data is available from the superordinated control unit.	1 1 x x x x X 1	1 1 x x 1 x X 0
5	When ASCII unit has taken the user data from the receive buffer, ASCII sets bit 0 in the handshake send byte and acknowledges		1 1 x x 1 x X 1
6	The handshake send byte of the gateway is transferred to the superordinated control unit.		1 1 x x 1 x X 1
7	The superordinated control unit recognizes that handshake bit 0 in the send and receive buffer is equal → ASCII has received the data and is ready to receive new user data	1 1 x x x x X 1	1 1 x x 1 x X 1
8	The superordinated control unit provides new user send data to ASCII and sets the handshake bit 0 to zero.	1 1 x x x x X 0	
9	The data is transferred to ASCII	1 1 x x x x X 0	
10	As long as bit 0 of the handshake in the send and receive buffer is unequal, it is not allowed for the superordinated control unit to send new user data to the gateway.	1 1 x x x x X 0	1 1 x x 1 x X 1
11	ASCII recognizes that bit 0 of the handshake in the send and receive buffer is unequal, that new user data is available.	1 1 x x x x X 0	1 1 x x 1 x X 1

© Hilscher, 2010-2011

Step	Action: **The superordinated control unit sends data to ASCII**	**Handshake-send byte of the superordinated control unit / receive byte of the gateway** 7 6 5 4 3 2 1 0	**Handshake-receive byte of the superordinated control unit / send byte of the gateway** 7 6 5 4 3 2 1 0
12	When ASCII has taken the user data from the receive buffer, the superordinated control unit sets bit 0 to zero in the handshake send byte and acknowledges		1 1 x x 1 x X 0
13	The handshake send byte of ASCII is transferred to the superordinated control unit.		1 1 x x 1 x X 0
14	The superordinated control unit recognizes that handshake bit 0 in the send and receive buffer is equal → ASCII has received the data and is ready to receive new user data		1 1 x x 1 x X 0
15	Handshake bit 0 in the receive and send buffer has the same value. This is the same state as in step 0. Now this procedure can start from the beginning.	1 1 x x x x X 0	1 1 x x 1 x X 0

Table 10: ASCII – Synchronization Register: Send ASCII Telegram

2.3.3.2 ASCII to the superordinated Control Unit

The value of a bit marked with x does not matter.

Step	Action: **ASCII sends data to the superordinated control unit**	**Handshake-send byte of the superordinated control unit / receive byte of ASCII** 7 6 5 4 3 2 1 0	**Handshake-receive byte of the superordinated control unit / send byte of ASCII** 7 6 5 4 3 2 1 0
0	State before beginning to send. Handshake bit 1 in the receive and send buffer has the same value. → It is possible to send data to the superordinated control unit.	1 1 x x x x 0 X	1 1 x x 1 x 0 X
1	ASCII has provided user data to be sent and sets handshake bit 1.		1 1 x x 1 x 1 X
2	The data is transferred to the superordinated control unit		1 1 x x 1 x 1 X
3	As long as bit 1 of the handshake in the send and receive buffer is unequal, ASCII will not send new user data to the superordinated control unit.	1 1 x x x x 0 X	1 1 x x 1 x 1 X
4	The superordinated control unit recognizes that bit 1 of the handshake in the send and receive buffer is unequal, that new user data from ASCII is available.	1 1 0 0 0 0 0 X	1 1 x x 1 x 1 X
5	When the superordinated control unit has taken the user data from the receive buffer, the superordinated control unit sets bit 1 in the handshake send byte and acknowledges	1 1 0 0 0 0 1 X	
6	The handshake send byte of the superordinated control unit is transferred to ASCII.	1 1 0 0 0 0 1 X	
7	ASCII recognizes that handshake bit 1 in the send and receive buffer is equal → The superordinated control unit has received the data and is ready to receive new user data	1 1 0 0 0 0 1 X	1 1 x x 1 x 1 X
8	ASCII provides new user send data and sets the handshake bit 1 to zero.		1 1 x x 1 x 0 X
9	The data is transferred to the superordinated control unit		1 1 x x 1 x 0 X
10	As long as bit 1 of the handshake in the send and receive buffer is	1 1 0 0 0 0 1 X	1 1 x x 1 x 0 X

© Hilscher, 2010-2011

Step	Action: ASCII sends data to the superordinated control unit	Handshake-send byte of the superordinated control unit / receive byte of ASCII 7 6 5 4 3 2 1 0	Handshake-receive byte of the superordinated control unit / send byte of ASCII 7 6 5 4 3 2 1 0
	unequal, ASCII is not allowed to send new user data to the superordinated control unit.		
11	The superordinated control unit recognizes that bit 1 of the handshake in the send and receive buffer is unequal, that new user data from ASCII is available.	1 1 0 0 0 0 1 X	1 1 x x 1 x 0 X
12	When the superordinated control unit has taken the user data from the receive buffer, the superordinated control unit sets bit 1 to zero in the handshake send byte and acknowledges	1 1 0 0 0 0 0 X	1 1 x x 1 x 0 X
13	The handshake send byte of the superordinated control unit is transferred to ASCII.	1 1 0 0 0 0 0 X	1 1 x x 1 x 0 X
14	ASCII recognizes that handshake bit 1 in the send and receive buffer is equal → The superordinated control unit has received the data and is ready to receive new user data		
15	Handshake bit 1 in the receive and send buffer has the same value. This is the same state as in step 0. Now this procedure can start from the beginning.	1 1 x x x x 0 X	1 1 x x 1 x 0 X

Table 11: ASCII – Synchronization Register: Receive ASCII Telegram

© Hilscher, 2010-2011

3 Error Codes

Error Codes	Name and Description
0000 0000 hex 0 dez	TLR_S_OK No error
C000 0007 hex 3221225479 dez	TLR_E_INVALID_PACKET_LEN Invalid packet length
C000 0180 hex 3221225857 dez	TLR_E_BUS_OFF The ASCII stack has been stopped by the host application.
C07E 0001 hex 3229483009 dez	TLR_E_ASCII_COMMAND_INVALID Invalid command received.
C07E 0002 hex 3229483010 dez	TLR_E_ASCII_STACK_DATA_SIZE_INVALID Invalid data length
C07E 0003 hex 3229483011 dez	TLR_E_ASCII_STACK_BUSY The serial line is busy. Any sending or receiving is in progress
C07E 0004 hex 3229483012 dez	TLR_E_ASCII_STACK_PACKET_TOO_LONG ASCII sends: Requested packet is too long for sending ASCII receives: The received telegram is longer than expected.
C07E 0005 hex 3229483013 dez	TLR_E_ASCII_STACK_DATA_OVERLAPPED ASCII receives. A new telegram has been received via the serial line before the previous was acknowledged by the superordinated control unit. The previous data in the internal buffer of the ASCII protocol has been overwritten.
C07E 0006 hex 3229483014 dez	TLR_E_ASCII_STACK_RESPONSE_TIMEOUT Answer time limit has been exceeded.
C07E 0007 hex 3229483015 dez	TLR_E_ASCII_STACK_WAITING_RESPONSE The ASCII protocol expects a response via the serial line, e. g. a receive of data from the serial interface is expected.
C07E 0008 hex 3229483016 dez	TLR_E_ASCII_STACK_LED_NOT_SUPPORTED LED not supported
C07E 0009 hex 3229483017 dez	TLR_E_ASCII_STACK_MSG_MODE_DISABLED Message mode is deactivated

Table 12: Error Codes ASCII

© Hilscher, 2010-2011

4 Appendix

4.1 List of Figures

Figure 1: Principal Mapping of the Data between the Protocols 9

4.2 List of Tables

Table 1: List of Revisions 3
Table 2: Reference to Firmware 4
Table 3: Reference to Software 4
Table 4: ASCII – Structure for Output – Data from the Control Unit 10
Table 5: ASCII – Structure for Input – Data to the Control Unit 10
Table 6: Relationship between send/receive ASCII Telegram and ASCII Operating Mode 12
Table 7: ASCII – Synchronization Register to ASCII 13
Table 8: ASCII – Synchronization Register to the Superordinated Control Unit 14
Table 9: ASCII – Initializing of the Communication 15
Table 10: ASCII – Synchronization Register: Send ASCII Telegram 17
Table 11: ASCII – Synchronization Register: Receive ASCII Telegram 18
Table 12: Error Codes ASCII 19

© Hilscher, 2010-2011

4.3 Contacts

Headquarters

Germany
Hilscher Gesellschaft für
Systemautomation mbH
Rheinstrasse 15
65795 Hattersheim
Phone: +49 (0) 6190 9907-0
Fax: +49 (0) 6190 9907-50
E-Mail: info@hilscher.com

Support
Phone: +49 (0) 6190 9907-99
E-Mail: de.support@hilscher.com

Subsidiaries

China
Hilscher Systemautomation (Shanghai) Co. Ltd.
200010 Shanghai
Phone: +86 (0) 21-6355-5161
E-Mail: info@hilscher.cn

Support
Phone: +86 (0) 21-6355-5161
E-Mail: cn.support@hilscher.com

France
Hilscher France S.a.r.l.
69500 Bron
Phone: +33 (0) 4 72 37 98 40
E-Mail: info@hilscher.fr

Support
Phone: +33 (0) 4 72 37 98 40
E-Mail: fr.support@hilscher.com

India
Hilscher India Pvt. Ltd.
New Delhi - 110 025
Phone: +91 11 40515640
E-Mail: info@hilscher.in

Italy
Hilscher Italia srl
20090 Vimodrone (MI)
Phone: +39 02 25007068
E-Mail: info@hilscher.it

Support
Phone: +39 02 25007068
E-Mail: it.support@hilscher.com

Japan
Hilscher Japan KK
Tokyo, 160-0022
Phone: +81 (0) 3-5362-0521
E-Mail: info@hilscher.jp

Support
Phone: +81 (0) 3-5362-0521
E-Mail: jp.support@hilscher.com

Korea
Hilscher Korea Inc.
Suwon, 443-734
Phone: +82 (0) 31-695-5515
E-Mail: info@hilscher.kr

Switzerland
Hilscher Swiss GmbH
4500 Solothurn
Phone: +41 (0) 32 623 6633
E-Mail: info@hilscher.ch

Support
Phone: +49 (0) 6190 9907-99
E-Mail: ch.support@hilscher.com

USA
Hilscher North America, Inc.
Lisle, IL 60532
Phone: +1 630-505-5301
E-Mail: info@hilscher.us

Support
Phone: +1 630-505-5301
E-Mail: us.support@hilscher.com

© Hilscher, 2010-2011